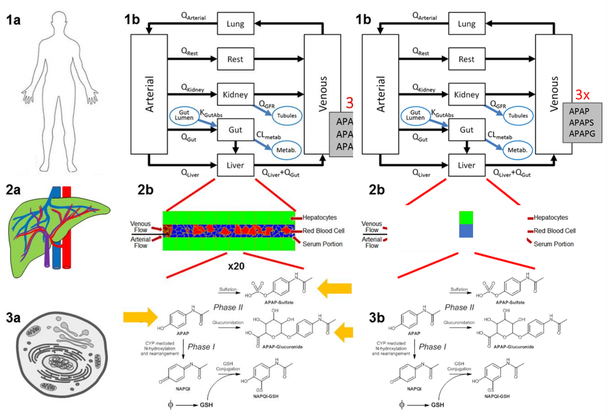
Parameter Fitting in a multiscale model: Parameter Scanning vs. Particle Swarm Optimization

James P. Sluka*, Anwar Shaikh, James A Glazier

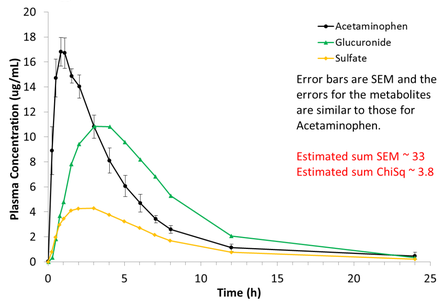
Biocomplexity Institute, Intelligent Systems Engineering, School of Informatics, Computing and Engineering
Indiana University, Bloomington, Indiana

Introduction: As computational biology models get more complex, for example in the case of multi-scale models, the use of available in vivo data for parameter estimation and model fitting tasks becomes increasingly difficult. Multi-scale models typically contain multiple parameters at each modeled scale resulting in a large parameter space that needs to be explored to identify possible solutions consistent with available experimental data. In our previously reported multi-scale model of Acetaminophen (*APAP*) pharmacokinetics there are more than 30 parameters for which direct experimental measurement are not available. Given the vast parameter space, and assuming there exist multiple local minima in any fitted model, coupled with the high likelihood of parameter interaction, there is a need for tools capable of efficiently exploring a large parameter space. Multiple approaches are possible for this challenge, including exhaustive scanning of parameter space, gradient based optimization methods, or stochastic methods such as simulated annealing, evolutionary algorithms or particle swarms. Our liver-centric APAP model in CC3D includes a whole-body PBPK model in SBML, a tissue scale model of blood and liver hepatocytes, and a subcellular metabolic model (in SBML). Here we report the use of two parameter exploration and optimization techniques for this model. Previous releases of CC3D included a parameter scanning module that exhaustive scans a user defined set or parameters and values. Here we describe an application of a particle swarm-based approach that can sample a larger number of parameters across broader ranges in a computationally efficient way. As a test case we have explored fitting ~10 adjustable parameters in the multi-scale APAP model. In parameter scanning we used a coarse scan followed by a finer scan with narrower parameter ranges. In the particle swarm approach we explored a much larger parameter space. Both the parameter scan and particle swarm approaches identified plausibly good sets of parameters consistent with the human ADME data.

1. Test Model: We have previously published a multiscale model of APAP distribution, metabolism and clearance [1](upper left). That model includes more than 30 adjustable parameters and takes about 30 CPU hours to simulate 12 hours. It was extremely difficult to fit available experimental data, and even with access to a super computer, the parameter fitting process was slow and incomplete. To use this APAP model as a test case for parameter scanning and particle swarm optimization we have created a simplified model (upper right). The reduced model includes both the whole-body PBPK models and the subcellular reaction kinetics model but greatly simplifies the multicell scale of the model by eliminating the blood flow simulation. The resulting simulation executes in ~40 seconds compared to the 30 hours of the full model.



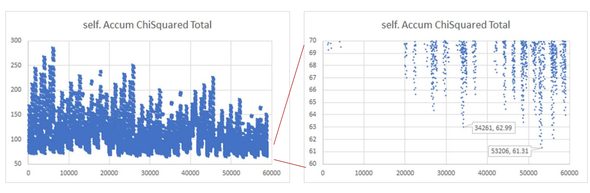
2. Human ADME Data: As a test case we use published human ADME data from 9 Caucasian males dosed with 1.4g (oral) of APAP [2] (right). The data set consist of blood serum concentration versus time data for APAP and its two main metabolites; APAP-Glucuronide and APAP-Sulfate.



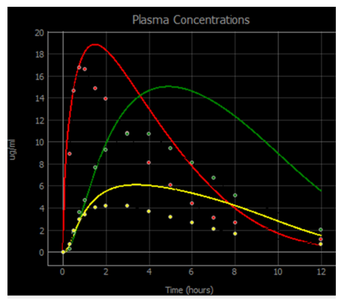
3. Parameter Scanning: CC3D already includes a parameter sweep tool that automatically generates and runs a set of simulation based on exhaustive enumeration across a set of user defined variable values. We performed two scans; an initial "coarse" scan centered around our previously published best parameter set followed by a finer scan centered at the best solution form the coarse scan.

Coarse Parameter Scan: The coarse scan results are shown below. The parameter list with range (max/min value) for the 10 scanned parameters (3 values/parameter) are below left. The results of all 59,049 (3^{10}) simulations are shown below center. The bottom center table shows the best parameters sets and the bottom right graph shows the results for the best parameter set (lines) overlaid on the human data (dots) for APAP (red), APAP-Glucuronide (green) and APAP-Sulfate (yellow).

```
<ParameterScan version="3.7.0">
  <OutputDirectory>Multi_simple_liver_pscan6_ParameterScan</OutputDirectory>
  <ParameterList Resource="Simulation/Multi_simple_liver_parameters.py">
    <Parameter CurrentIteration="0" Name="pbpk_kGutabs" Type="PYTHON_GLOBAL" ValueTy
      <Values>1,1,1.4,1.7</Values>
    </Parameter>
    <Parameter CurrentIteration="1" Name="cc3d_Km_AT_APAP" Type="PYTHON_GLOBAL" Val
      <Values>0.005,0.01,0.015</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_Vmax_AT_APAP" Type="PYTHON_GLOBAL" 
      <Values>0.0002,0.0003,0.0004</Values>
    </Parameter>
    <Parameter CurrentIteration="1" Name="cc3d_k_PD_S2H" Type="PYTHON_GLOBAL" Value1
      <Values>0.0005,0.001,0.0015</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_k_AT_APAPG" Type="PYTHON_GLOBAL" Val
      <Values>0.0003,0.00045,0.0006</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_k_AT_APAPS" Type="PYTHON_GLOBAL" Valu
      <Values>0.0012,0.0019,0.0026</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Km_GLUC" Type="PYTHON_GLOBAL" ValueTyp
      <Values>1.0,1.5,2.0</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Km_SULF" Type="PYTHON_GLOBAL" ValueType
      <Values>2.2,3.0,4</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Vmax_GLUC" Type="PYTHON_GLOBAL" ValueT
      <Values>0.0006,0.0009,0.0012</Values>
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Vmax_SULF" Type="PYTHON_GLOBAL" ValueTy
      <Values>0.0001,0.00015,0.0002</Values>
    </Parameter>
  </ParameterList>
</ParameterScan>
```

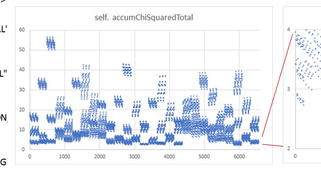


cc3d_Km_AT_APAP	cc3d_Vmax_AT_APAP	cc3d_k_AT_APAPG	cc3d_k_AT_APAPS	cc3d_k_PD_S2H	pbpk_kGutabs	sc_Km_GLUC	sc_Km_SULF	sc_Vmax_GLUC	sc_Vmax_SULF	self.coarseTotal	self.fineTotal	self.totalChiSquared	Run#
0.015	0.0003	0.0006	0.0026	0.0015	1.4	1	0.2	0.0012	0.0002	87.92	3.10	61.31	53206
0.015	0.0003	0.0006	0.0019	0.0015	1.4	1	0.2	0.0012	0.0002	88.13	3.10	61.60	52963
0.015	0.0003	0.0006	0.0026	0.0015	1.1	1	0.2	0.0012	0.0002	87.37	3.05	61.91	53205
0.015	0.0004	0.0006	0.0026	0.0015	1.4	1.5	0.3	0.0012	0.0002	87.25	3.10	62.08	56131
0.015	0.0003	0.0006	0.0019	0.0015	1.1	1	0.2	0.0012	0.0002	87.77	3.06	62.25	52962
0.015	0.0003	0.0006	0.0012	0.0015	1.4	1	0.2	0.0012	0.0002	88.84	3.10	62.31	52720
0.015	0.0004	0.0006	0.0019	0.0015	1.4	1.5	0.3	0.0012	0.0002	87.46	3.10	62.41	55888
0.015	0.0003	0.00045	0.0026	0.0015	1.4	1	0.2	0.0012	0.0002	89.25	3.11	62.48	53125
0.015	0.0003	0.00045	0.0019	0.0015	1.4	1	0.2	0.0012	0.0002	89.47	3.11	62.77	52882
0.015	0.0004	0.0006	0.0026	0.0015	1.1	1.5	0.3	0.0012	0.0002	87.68	3.07	62.79	56130

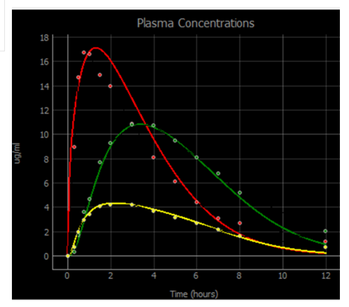


Fine Parameter Scan: Starting at the best solution from the coarse scan, we re-centered and narrowed the regions, omitted some low-sensitivity parameters and added two new parameters to the scan. This scan included 6561 (3^8) simulations. The parameters scanned and results are shown below as done for the coarse scan.

```
<ParameterScan version="3.7.0">
  <OutputDirectory>Multi_simple_liver_pscan7_ParameterScan</OutputDirectory>
  <ParameterList Resource="Simulation/Multi_simple_liver_parameters.py">
    <Parameter CurrentIteration="0" Name="pbpk_QgfrG" Type="PYTHON_GLOBAL"
      <Values>20.0,22.0,24.0</Values>                                  1.2x new
    </Parameter>
    <Parameter CurrentIteration="0" Name="pbpk_QgfrS" Type="PYTHON_GLOBAL"
      <Values>17.8,19.8,21.8</Values>                                  1.2x new
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_Vmax_AT_APAP" Type="PYTHON
      <Values>0.00046,0.00066,0.00086</Values>                         1.9x
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_k_AT_APAPG" Type="PYTHON_G
      <Values>0.000395,0.000495,0.000595</Values>                      1.5x
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_k_AT_APAPS" Type="PYTHON_GL
      <Values>0.0045,0.0055,0.0065</Values>                            1.4x
    </Parameter>
    <Parameter CurrentIteration="0" Name="cc3d_k_PD_H2S" Type="PYTHON_GLOI
      <Values>0.00125,0.00165,0.00205</Values>                         1.6x
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Vmax_GLUC" Type="PYTHON_GLOE
      <Values>0.00102,0.00132,0.00162</Values>                         1.6x
    </Parameter>
    <Parameter CurrentIteration="0" Name="sc_Vmax_SULF" Type="PYTHON_GLOB.
      <Values>0.000115,0.000165,0.000215</Values>                      1.9x
    </Parameter>
  </ParameterList>
</ParameterScan>
```

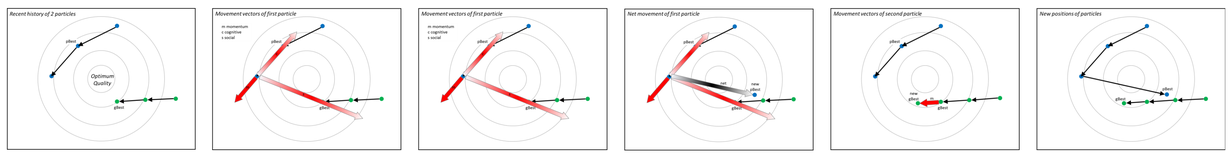


cc3d_Vmax_AT_APAP	cc3d_k_AT_APAPG	cc3d_k_AT_APAPS	cc3d_k_PD_H2S	pbpk_QgfrG	pbpk_QgfrS	sc_Vmax_GLUC	sc_Vmax_SULF	CumTotErr	RMSeTotal	accumChiSquaredTotal	Run#
0.00046	0.000495	0.0045	0.00125	20	21.8	0.00162	0.000215	16.55	0.693	2.17	5865
0.00046	0.000495	0.0055	0.00125	20	21.8	0.00162	0.000215	16.68	0.694	2.18	5946
0.00046	0.000495	0.0065	0.00125	20	21.8	0.00162	0.000215	16.80	0.694	2.20	6027
0.00046	0.000595	0.0045	0.00125	20	21.8	0.00162	0.000215	17.49	0.708	2.23	5892
0.00046	0.000595	0.0055	0.00125	20	21.8	0.00162	0.000215	17.62	0.708	2.24	5973
0.00046	0.000595	0.0065	0.00125	20	21.8	0.00162	0.000215	17.74	0.709	2.26	6054
0.00066	0.000495	0.0065	0.00205	24	17.8	0.00162	0.000165	17.40	0.685	2.34	4331
0.00066	0.000495	0.0055	0.00205	24	17.8	0.00162	0.000165	17.51	0.685	2.35	4250
0.00066	0.000495	0.0065	0.00205	22	17.8	0.00162	0.000165	18.69	0.700	2.36	4330
0.00066	0.000495	0.0065	0.00165	22	19.8	0.00132	0.000165	17.04	0.661	2.36	3361
0.00066	0.000495	0.0055	0.00165	22	19.8	0.00132	0.000165	17.08	0.662	2.36	3280



Parameter Scan Conclusion: The parameter scan tool in CC3D works well, though for parameter fitting it quickly becomes intractable if the number of adjustable parameters is more than a few. Alternative approaches are needed to effectively explore the large parameter space.

4. Particle Swarm (*PS*): PS [3] is a semi-random exploration of parameter space that optimizes by iteratively trying to improve candidate solutions with regard to a given measure of quality. A "swarm" of candidate solutions ("particles") moves around the search-space according to a simple protocol that modifies the particle's position and velocity. Each particle's movement is influenced by its previous best position and by the best position for the entire swarm (see below). A PS does not use any gradient information beyond ranking of current versus previous best positions. PS optimization makes few assumptions about the nature of the parameter space being searched or of the model used to evaluate the quality of a particular solution. PS optimization is often an effective way of searching large parameter spaces.



Particle Swarm Workflow: The PS workflow consist of gathering the necessary files and defining the parameters to be scanned along with their ranges. We have developed a python program that carries out the PS optimization using a CC3D simulation. It evaluates the individual particle's quality based on the ability to reproduce the human APAP blood data. Note that our PS optimizer is adaptive and will work with any other quality metric as well. The PS code uses the CC3D simulation and list of parameters (and their ranges) to create, execute and analyze a series of simulations on a *nix computing cluster. We use the *Slurm* workload manager to partition the jobs across the available compute nodes on a cluster. Since the stepwise movement of the individual particles is independent of the behavior of other particles the entire computation can be carried in parallel.

Typical Parameter Scan: As a representative PS optimization we used the same model as in the parameter scans. We allowed 12 parameters to vary over much wider ranges (below) than in the parameter scans. We used 4 swarms of 30 particles each and a total of 200 iterations requiring 24,000 simulations (4·30·200). Note that the product of the parameter ranges used for the PS is approximately 10^{19} times larger than that in the coarse parameter scan. The PS parameters and results are shown below.

```
pbpk_kGutabs,0.8,2.0                    # 1/hr          2.5x

cc3d_k_AT_APAPG,0.00005,0.005           # 1/s           100x
cc3d_k_AT_APAPS,0.0005,0.05             # 1/s           100x

cc3d_Km_AT_APAP,0.00250,0.25            # mmol/L        100x
cc3d_Vmax_AT_APAP,0.00005,0.005         # mmol/L/s      100x

sc_Km_GLUC,0.12,12.                     # mmol/L        100x
sc_Vmax_GLUC,0.00015,0.015              # mmol/L/s      100x

sc_Km_SULF,0.0200,2.                    # mmol/L        100x
sc_Vmax_SULF,0.00002,0.002              # mmol/L/s      100x

cc3d_k_PD_S2H,0.0001,0.01               # 1/s           100x

pbpk_QgfrG,2.0,200.0                    # L/hr          100x
pbpk_QgfrS,2.0,200.0                    # L/hr          100x
```


Swarm	pbpk_kGutabs	cc3d_k_AT_APAPG	cc3d_k_AT_APAPS	cc3d_Km_AT_APAP	cc3d_Vmax_AT_APAP	sc_Km_GLUC	sc_Vmax_GLUC	sc_Km_SULF	sc_Vmax_SULF	cc3d_k_PD_S2H	pbpk_QgfrG	pbpk_QgfrS	bestE:
0	1.4338	0.0005	0.0006	0.0897	0.002	2.2662	0.0022	1.7091	0.0015	0.0056	13.4398	32.9585	2.6092
1	1.7450	0.0013	0.0317	0.1531	0.003	10.9219	0.0018	0.6031	0.0002	0.0008	11.1633	38.2368	1.9975
2	1.4614	0.0004	0.0007	0.0796	0.0024	7.3969	0.0085	1.981	0.0018	0.0100	15.3846	27.3316	2.5505
3	1.4353	0.001	0.0005	0.2466	0.002	10.2321	0.0082	0.0915	0.0003	0.0021	6.0175	83.4308	2.0809

Particle Swarm Conclusion: Overall, the PS optimization found both a better optimal solution and more alternate solutions, and did so in less time (compute and wall clock) than the parameter scan.

CONCLUSIONS: For the particle swarm approach, a similar set of parameters were explored over a 10^{15} wider range of values for each parameter, requiring 24,000 simulations. The particle swarm approach identified multiple plausible parameter sets consistent with the human ADME data. Overall, the particle swarm proved remarkably effective at sampling the large parameter space and was found to be a more efficient approach than parameter scanning.

References

[1] Sluka JP, *et al.* A Liver-Centric Multiscale Modeling Framework for Xenobiotics. PLoS One. 11(9):e0162428. doi: 10.1371/ journal. pone. 0162428 (2016).

[2] Critchley, J.A.J.H. *et al.* Differences in the single-oral-dose pharmacokinetics and urinary excretion of paracetamol and its conjugates between Hong Kong Chinese and Caucasian subjects. J. Clin. Pharm. Ther. 30, 179–184 (2005).

[3] Bonyadi, M.R., Michalewicz, Z. "Particle swarm optimization for single objective continuous space problems: a review". Evolutionary Computation. 25 (1): 1–54. doi:10.1162/EVCO_r_00180 (2017).

IMAG/MSM Meeting, Bethesda MD, March 2018